

Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Presently amended) A hardware-based multithreaded processor comprising:
a plurality of microengines, each of the microengines comprising:
a control store;
controller logic;
context event switching logic; and
an execution box data path including an arithmetic logic unit (ALU) and a general
purpose register set, the ALU performing functions in response to instructions, one of the
instructions causing the ALU ~~A computer instruction comprises:~~
~~a command instruction~~ to issue a memory reference to an address in a memory shared
among threads executing in the microengines ~~microprocessors~~ while a context of a thread is
inactive.
2. (Presently amended) The processor instruction of claim 1 wherein the ~~command~~ instruction
comprises a command field that sets or clears user-specified bits in a longword.
3. (Presently amended) The processor instruction of claim 1 wherein the ~~command~~ instruction
comprises a command field that reads from the address to a transfer register associated with the
microengines.
4. (Presently amended) The processor instruction of claim 1 wherein the ~~command~~ instruction
comprises a command field that locks the memory and then reads the memory.
5. (Presently amended) The processor instruction of claim 1 wherein the ~~command~~ instruction
comprises a command field that writes to the memory from a transfer register associated with the
microengines.
6. (Presently amended) The processor instruction of claim 1 wherein the ~~command~~ instruction
comprises a command field that writes to the address and unlocks the address.

7. (Presently amended) The processor instruction of claim 1 wherein the ~~command~~ instruction comprises a command field that pushes a list element specified by the address onto a specified stack.
8. (Presently amended) The processor instruction of claim 1 wherein the ~~command~~ instruction comprises a command field that pops a list element specified by the address from a specified stack.
9. (Presently amended) The processor instruction of claim 1 further comprising:
a transfer register specified as a parameter in the instruction.
10. (Presently amended) The processor instruction of claim 1 wherein the instruction further ~~comprising~~ comprises:
a first source operand field; and
a second source operand field.
11. (Presently amended) The processor instruction of claim 10 wherein the first source operand and the second source operand are context-relative registers.
12. (Presently amended) The processor instruction of claim 10 wherein the first source operand and the second source operand are 5-bit intermediate data ranging from +31 to 0.
13. (Presently amended) The processor instruction of claim 1 further comprising a reference count field specified as a parameter in the instruction.
14. (Presently amended) The processor instruction of claim 13 wherein the reference count field specifies a number of contiguous longwords in the memory to be referenced.
15. (Presently amended) The processor instruction of claim 1 further comprising a queue number as a parameter in the instruction.
16. (Presently amended) The processor instruction of claim 15 wherein the queue number specifies one of eight push/pop queues.
17. (Presently amended) The processor instruction of claim 1 further comprising a bit operand as a parameter in the instruction.
18. (Presently amended) The processor instruction of claim 17 wherein the bit operand sets or clear bits at an address using a specified bit mask.
19. (Presently amended) The processor instruction of claim 1 further comprising:
an optional token that is set by a programmer in the instruction.

20. (Presently amended) The processor instruction of claim 19 wherein the optional token causes the instruction to signal a corresponding micro-engine/thread pair that is sourcing or sinking memory data when complete.

21. (Presently amended) The processor instruction of claim 19 wherein the optional token swaps out a context of a current thread execution to let another thread context execute.

22. (Presently amended) The processor instruction of claim 19 wherein the optional token swaps out a current context thread after execution of one instruction.

23. (Presently amended) The processor instruction of claim 19 wherein the optional token places a memory reference into an ordered queue.

24. (Presently amended) The processor instruction of claim 19 wherein the optional token places a memory reference into a priority queue.

25. (Presently amended) The processor instruction of claim 19 wherein the optional token optimizes memory bandwidth by placing the memory reference into a read or ordered queue.

26. (Presently amended) The processor instruction of claim 19 wherein the optional token indicates overriding qualifiers.

27. (Presently amended) The processor instruction of claim 1 wherein the memory is a synchronous dynamic random access memory (SDRAM).

28. (Presently amended) The processor instruction of claim 1 wherein the memory is a synchronous random access memory (SRAM).

29. (Presently amended) The processor instruction of claim 1 wherein the memory is a scratch pad memory.

30. (Original) A method of operating a processor comprising:
issuing a command to a memory shared among threads executing in microprocessors, each thread having an associated context; and
inactivating the context of the thread issuing the command while the command is executing.

31. (Original) The method of claim 30 wherein the command comprises:
setting user-specified bits in a longword.

32. (Original) The method of claim 30 wherein the command comprises:
clearing user-specified bits in a longword.

33. (Original) The method of claim 30 further comprising:
providing an address in the memory to affect a change.
34. (Original) The method of claim 33 wherein the command comprises:
locking the memory.
35. (Original) The method of claim 34 wherein the command further comprises:
reading from the address to a transfer register associated with the microprocessors.
36. (Original) The method of claim 33 further comprising:
unlocking the memory; and
writing to the address from a transfer register associated with the microprocessors.